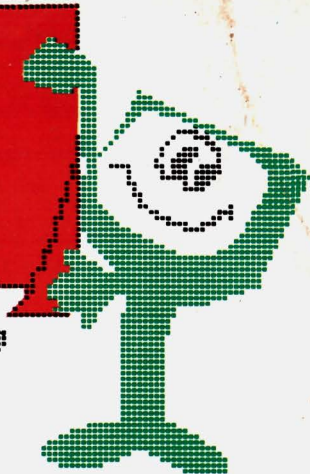
20 VIDEOLEZIONI DI BASIC
PER IMPARARE COL VIC 20

**GRUPPO
EDITORIALE
JACKSON**

- Inconvenienti del computer
- Manutenzione: tastiera, unità centrale, televisione, memoria di massa, stampante
- Interpreti e compilatori
- Funzioni numeriche
- EXP, LOG, DEF FN
- Usare e definire le funzioni
- Videogioco n° 16

16

COMMODORE VIC 20

VIDEO BASIC VIC 20
Pubblicazione quattordicinale
edita dal Gruppo Editoriale Jackson
Direttore Responsabile:
Giampietro Zanga
Direttore e Coordinatore
Editoriale: Roberto Pancaldi
Autore: Softidea - Via Indipendenza 88 - Como
Redazione software:
Francesco Franceschini, Enrico Braglia,
Fabio Calanca
Segretaria di Redazione:
Marta Menegardo
Progetto grafico:
Studio Nuovaidea - Via Longhi 16 - Milano
Impaginazione:
Silvana Corbelli
Illustrazioni:
Cinzia Ferrari, Silvano Scolari
Fotografie:
Marcello Longhini
Distribuzione: SODIP
Via Zuretti, 12 - Milano
Fotocomposizione: Lineacomp S.r.l.
Via Rosellini, 12 - Milano
Stampa: Grafika '78
Via Trieste, 20 - Pioltello (MI)
Direzione e Redazione:
Via Rosellini, 12 - 20124 Milano
Tel. 02/6880951/5

Autorizzazione alla pubblicazione Tribunale di Milano n° 422 del 22-9-1984
Spedizione in abbonamento postale Gruppo II/70 (autorizzazione della Direzione Provinciale delle PPTT di Milano).
Prezzo del fascicolo L. 8.000
Abbonamento comprensivo di 5 raccoglitori L. 165.000
I versamenti vanno indirizzati a: Gruppo Editoriale Jackson S.r.l. - Via Rosellini, 12 20124 Milano, mediante emissione di assegno bancario o cartolina vaglia oppure utilizzando il c.c.p. n° 11666203
I numeri arretrati possono essere richiesti direttamente all'editore inviando L. 10.000 cdu. mediante assegno bancario o vaglia postale o francobolli.
Non vengono effettuate spedizioni contrassegno.

Gruppo Editoriale Jackson

SOMMARIO

Introduzione

Dopo ore e ore passate in solitudine assieme al proprio computer, può capitare di dimenticarsi che anche lui, in fondo, ... è umano. Come tutti gli uomini ha malanni e acciacchi vari. Le sue malattie? Fatta la necessaria trasposizione uomo-macchina, le artrosi (tasti), miopia (video), sordità (registratore, drive), tachicardia (chip). La cura? Una sola: prevenzione. Alcuni preziosi suggerimenti su come trattarlo, poi possono migliorare e di molto, la vita. Ogni tanto, infine, un buon check up ...

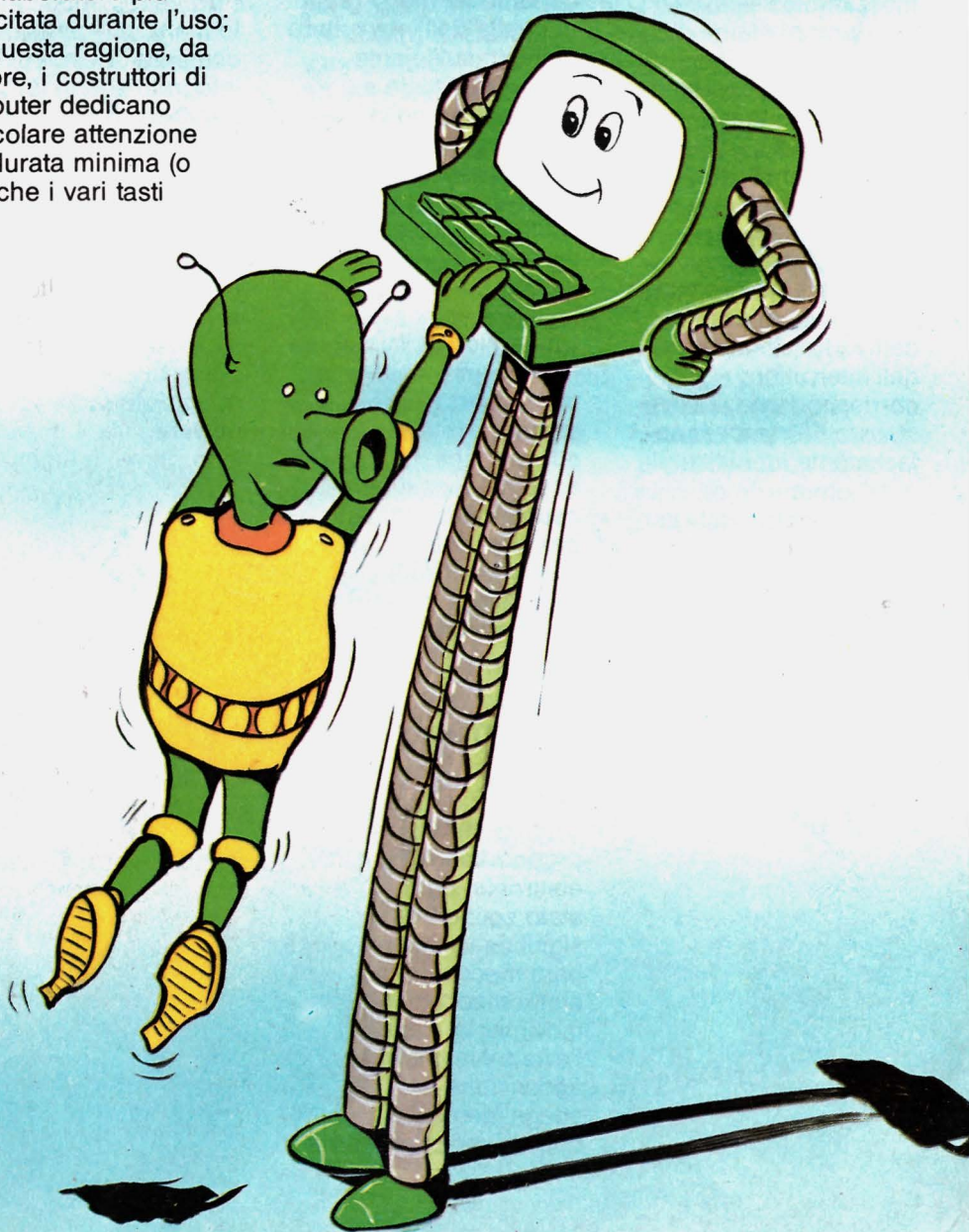
Inconvenienti e manutenzione

Nel corso degli anni i micro e i personal computer sono diventati estremamente affidabili, soprattutto grazie alla costante riduzione del numero di componenti elettronici che ne costituiscono le varie parti. L'affidabilità di un dispositivo elettronico (ed un elaboratore non sfugge certamente a questa regola) è infatti tanto maggiore quanto minore è la quantità di pezzi che lo compongono. In confronto ai loro predecessori i computer moderni sono quindi meno soggetti a guasti e più resistenti.

Tuttavia, come d'altra parte in qualsiasi *macchina costruita* dall'uomo, anche nei calcolatori (e relative periferiche) si possono talvolta riscontrare in difetti e malfunzionamenti, pregiudicando così l'affidabilità dell'intero sistema.
In genere, i guasti più comuni si localizzano nelle parti del computer prevalentemente soggette ad usura meccanica; per esempio: tastiera, spinotti di alimentazione o di connessione, stampante e drive.
Tali inconvenienti (a parte casi particolari e straordinari) fanno comunque parte del normale ciclo di vita della macchina: proprio come un'automobile può richiedere di tanto in tanto un'aggiustatina alle gomme o ai freni, anche per un computer è del tutto normale che si presenti la necessità di effettuare qualche manutenzione.
Oggi cercheremo quindi, attraverso un'analisi delle varie parti che costituiscono il tuo elaboratore, di esaminare le principali e più comuni cause di possibile cattivo funzionamento, suggerendone - quando *possibile - le soluzioni* più corrette ed adeguate da applicare per eliminarle.
La prima e più importante raccomandazione è comunque quella di leggere innanzitutto (e con attenzione) i manuali di utilizzo di tutti i dispositivi connessi al calcolatore: molte volte può *infatti* accadere di commettere un errore per scarsa conoscenza del funzionamento di un certo apparecchio. In tutti i manuali vengono inoltre forniti preziosi consigli ed informazioni sulle procedure di connessione, gestione e funzionamento delle varie unità. Spesso pochi minuti dedicati alla lettura dei manuali possono evitare parecchi giorni di attesa (oltre al costo in denaro) per le eventuali riparazioni conseguenti al non aver rispettato le corrette procedure operative.

La tastiera posta troppo in alto o troppo in basso costringe l'operatore ad una posizione scorretta e innaturale.

La tastiera

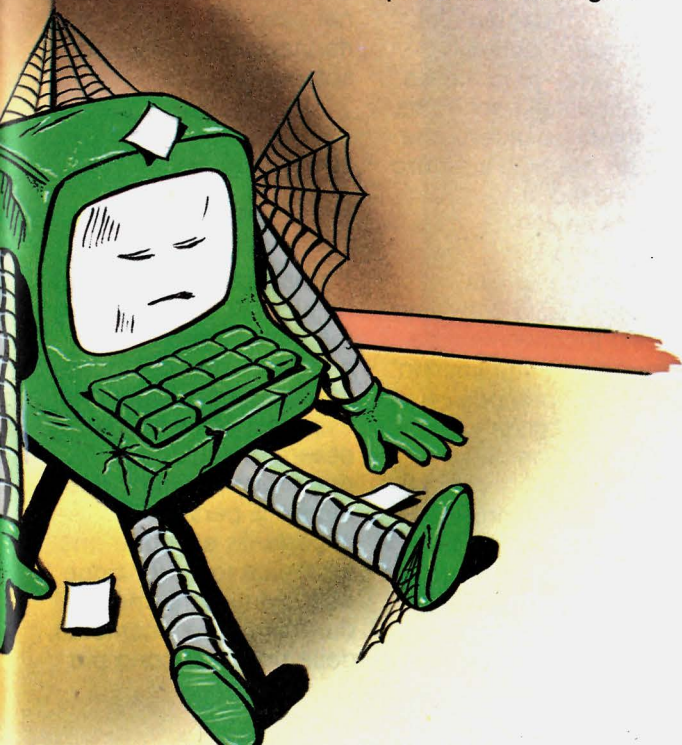
La tastiera è ovviamente una delle parti dell'elaboratore più sollecitata durante l'uso; per questa ragione, da sempre, i costruttori di computer dedicano particolare attenzione alla durata minima (o vita) che i vari tasti devono poter assicurare. Si è soliti indicare la durata della tastiera con il numero di battute

mediamente usufruibili prima di incappare in un qualsiasi malfunzionamento. Le moderne tastiere assicurano vite medie dell'ordine di milioni di battute, corrispondenti ad anni di uso normale dei vari tasti.
Gli inconvenienti che possono capitare a un generico tasto sono principalmente due: rottura o allentamento della molla di ritorno o cattivo funzionamento dell'interruttore corrispondente al tasto stesso. Entrambi sono facilmente identificabili ed eliminabili con la semplice sostituzione del pezzo difettoso.
La rottura della molla è certamente meno grave di quella dell'interruttore (è infatti sufficiente sollevare il tasto e sostituire la molla, mentre per cambiare l'interruttore si deve ricorrere all'uso del saldatore), tuttavia, sia in un caso che nell'altro, il guasto è riparabile con facilità e velocità.
Un nemico della tastiera (e in genere di tutte le parti elettroniche) è inoltre costituito dalla polvere, che accumulandosi può causare falsi contatti; per questo è buona norma coprire il calcolatore quando non è in funzione.

Il computer

Tutte le parti componenti il computer vero e proprio sono dispositivi elettronici del tipo "a stato solido". Ciò significa che non vi sono parti meccaniche o elettromeccaniche in movimento, tranne l'eventuale interruttore di accensione e spegnimento. La conseguenza più ovvia ed immediata di questo fatto è che in pratica non viene richiesta alcuna manutenzione.
Purché nessuno "metta le mani" all'interno del computer, i circuiti integrati, che in larga parte compongono l'elaboratore, possono funzionare (con le debite eccezioni) tranquillamente per mesi e mesi.
I nemici principali delle varie parti circuitali sono facilmente identificabili: polvere, urti e liquidi. Per quanto riguarda la polvere vale il discorso fatto prima: è bene

coprire il computer quando non lo si utilizza; la polvere potrebbe infatti (oltre a favorire i falsi contatti) impedire il normale scambio termico tra l'ambiente esterno e il computer, provocando surriscaldamenti (o addirittura bruciature) dei circuiti.

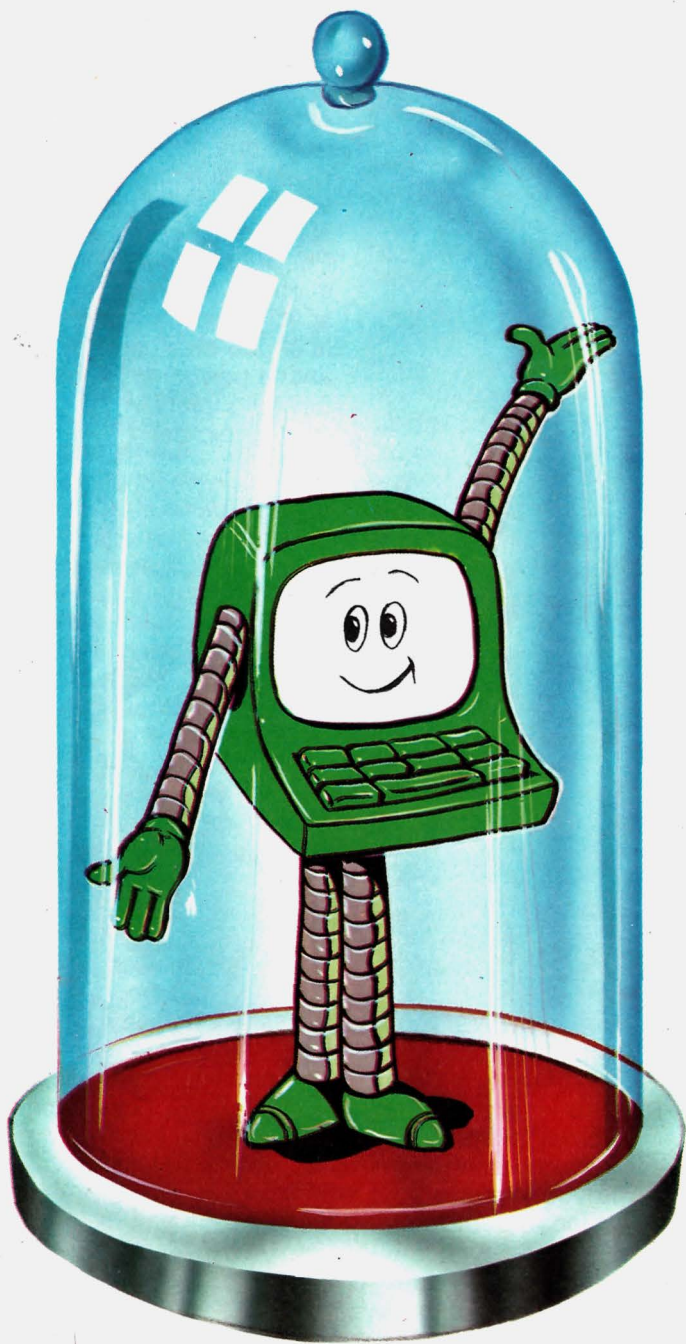
Particolari attenzioni vanno inoltre dedicate ad eventuali urti e vibrazioni. Un computer ben costruito è abbastanza protetto in questo senso, e tollera senza problemi spostamenti e trasporti, comprese le vibrazioni del bagagliaio di un'auto; bisogna comunque fare attenzione a non esagerare con gli sballottamenti, poiché a lungo andare, potrebbero insorgere problemi, dovuti magari all'allentamento di qualche vite o di qualche circuito interno. I personal computer sono quasi sempre molto robusti.

Un discorso a parte meritano invece i liquidi. Versato su un circuito elettrico un liquido gli è in ogni caso fatale; occorre quindi evitare di appoggiare bicchieri o bottiglie sopra qualsiasi dispositivo elettronico. Il pericolo è inoltre duplice: primo, qualunque liquido versato accidentalmente provocherà un corto circuito all'interno dell'elaboratore, con conseguenze imprevedibili, ma sicuramente disastrose. *Secondo, può insorgere* un rischio per l'incolumità delle persone che si dovessero trovare nelle vicinanze. Molto meglio bere in un'altra stanza!

Il televisore o il monitor

La manutenzione necessaria per conservare nel tempo l'efficienza e la qualità di un televisore o di un monitor è veramente minima e può essere riassunta in poche parole: tenerlo pulito e regolato correttamente. Anche nel caso dello schermo video si possono comunque considerare alcune elementari precauzioni. Primo: evitare - come al solito - l'accumulo di polvere. Secondo: posizionare l'unità in maniera che possa esservi un certo ricambio d'aria (questa regola è fondamentale). Terzo: non lasciare troppo a lungo la stessa immagine sullo schermo; può infatti accadere che i fosfori presenti sul retro dello stesso si schiariscano, imprimendo in permanenza quella immagine sullo schermo. Questo inconveniente è comunque molto limitato nei cinescopi moderni. Anche per l'unità video - in caso di mancato funzionamento - vale quanto detto a proposito del computer vero e proprio: non tentare assolutamente di effettuare riparazioni senza avere le opportune conoscenze teoriche e pratiche. È molto più facile per un inesperto peggiorare le cose invece di migliorarle. Inoltre - anche con la spina di alimentazione staccata dalla presa - all'interno del monitor o del televisore esistono delle parti sottoposte in permanenza ad un'elevata tensione e il cui contatto può risultare pericoloso per le persone. L'unica cosa da fare è controllare l'integrità del fusibile eventualmente inserito come protezione dell'unità e provvedere - nel caso fosse danneggiato - alla sua sostituzione.

Il drive e il registratore

Probabilmente i peggiori nemici dei dispositivi magnetici sono l'incuria e la distrazione; la quasi totalità dei problemi che essi accusano può infatti essere di solito ricondotta a queste due cause. È pertanto importante eseguire una periodica manutenzione delle varie parti che durante il funzionamento delle unità vengono più

utilizzate e quindi usurate.
Parleremo facendo *sempre riferimento* al drive: le stesse considerazioni valgono comunque anche per il registratore.
Per quanto riguarda il drive occorre verificare periodicamente che la calibrazione e

l'allineamento della testina siano corretti, cioè che la velocità di rotazione del motore e la posizione della testina rispetto al floppy disk siano identici a quanto stabilito dal costruttore. Per questa operazione ci si può rivolgere a uno specialista, ma volendo la si può effettuare anche per conto proprio. Esistono infatti in commercio dei programmi mediante i quali si possono controllare automaticamente sia lo stato della testina che la velocità del motore che pone in rotazione il dischetto. Naturalmente, qualora venissero rilevate alcune irregolarità è necessario ricorrere allo specialista per rimettere a posto le cose.

Il disallineamento della testina, soprattutto, è un inconveniente particolarmente subdolo. Usando il drive con la testina non allineata, di solito tutto sembra infatti

funzionare nel migliore dei modi: le registrazioni e le letture dei dischetti avvengono normalmente, senza alcun errore o anomalia. Se si tenta tuttavia di leggere un disco registrato da un'altra unità (o, viceversa, di leggere su un'altra unità un disco già registrato), l'operazione risulta impossibile.
Questo fatto è diretta conseguenza della posizione della testina che indica al drive la presenza di registrazioni diverse da quelle che si aspettava. In un certo senso il lettore di dischi riconosce solo le proprie registrazioni, "personalizzate" dal disallineamento della testina stessa.
Di solito un drive richiede di essere verificato una o due volte all'anno.
Ben più frequente deve essere invece la pulizia della testina. Essa va infatti pulita regolarmente, per eliminare la presenza di polvere e di particelle di ossido che su di essa si depositano in seguito al contatto con il disco. Il numero di puliture necessarie dipende dall'ambiente (più o meno polveroso) in cui il drive opera e dall'intensità di utilizzo alla quale viene sottoposto. A questo scopo sono disponibili speciali floppy disk, appositamente ideati per facilitare l'operazione.
Come regola generale, la testina del drive andrebbe pulita almeno una volta ogni tre mesi. La velocità e la semplicità dell'operazione consigliano tuttavia un intervento anche più frequente.
Occorre infine ricordarsi che anche i drive sono meccanismi sensibili; quando si sposta un drive si deve quindi, come al solito, cercare di evitare urti e vibrazioni.
Un drive ben curato può lavorare per anni senza alcun problema, purché si faccia sempre un minimo di attenzione (e di manutenzione).

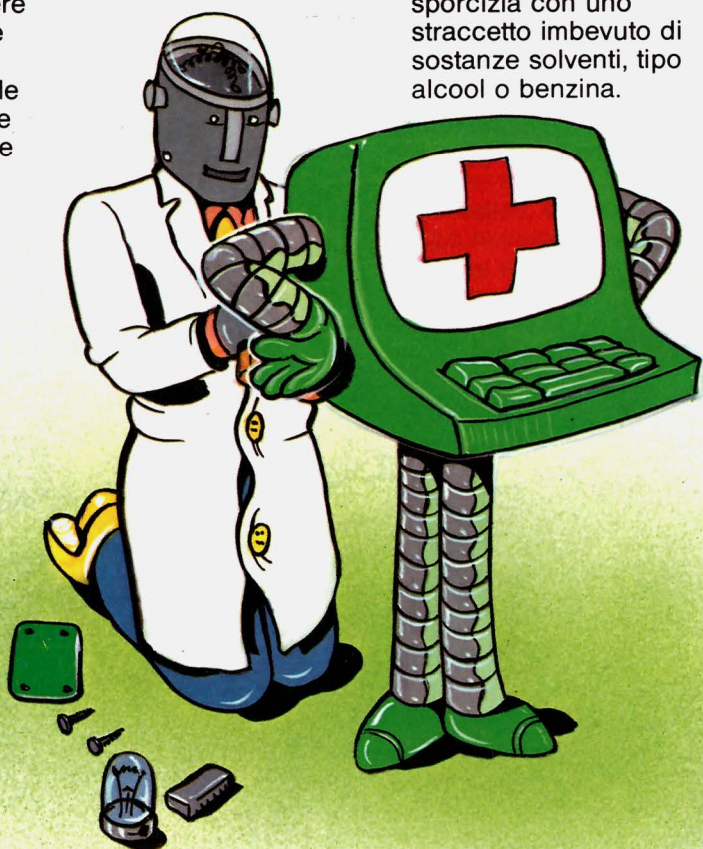
La stampante

In quasi tutti i sistemi la stampante è il dispositivo con il maggior numero di parti meccaniche, ed è quindi quello con maggiori probabilità di guasti, soprattutto se impiegato in modo anomalo o scorretto. Dato che una stampante ha tante parti mobili, risente particolarmente di un uso violento. Vediamo come comportarsi.
La stampante dovrebbe innanzitutto poggiare su un supporto stabile, per evitare vibrazioni che ne

disturbino il lavoro.
Dietro e davanti alla stampante ci dovrebbe inoltre essere uno spazio sufficiente per poter accedere alla carta in entrata e in uscita.
La carta va inserita correttamente nel meccanismo di trazione.
Può sembrare una considerazione evidente, però è facile infilare la carta storta, provocando talvolta il blocco della stampante. Occorre quindi imparare a rispettare sempre la disposizione della carta.
Un blocco della stessa è quasi sempre dovuto ad un'ostruzione fisica; di solito la carta si blocca perché i controlli sono regolati male, perché è stata inserita in modo scorretto o perché si sono utilizzati fogli non idonei.
Quando la carta si blocca il motore della stampante può bruciare: è quindi bene non assentarsi mai durante il funzionamento. Anche il nastro inchiostrato (nelle stampanti ad aghi e a margherita) deve essere sempre tenuto sotto controllo; bisogna saperlo inserire in modo corretto, facendo attenzione alle eventuali pieghe che possono sempre verificarsi durante il montaggio.
La testina della stampante è una delle parti maggiormente soggette ad usura, tanto che è quasi normale che a un certo punto si rompa. In questo caso l'unica cosa da fare è provvedere alla sostituzione.
Nel caso di guasto parziale o totale della stampante è bene eseguire nell'ordine queste operazioni:
— controllare che la posizione e l'inserimento dei cavi di alimentazione e di collegamento con il computer siano corretti;
— verificare tutti gli interruttori e i tasti di controllo;
— controllare lo stato del fusibile;
— controllare il meccanismo di stampa, il nastro e la carta;
— a questo punto, se le cose non sono ancora andate a posto, far controllare la stampante presso un centro di assistenza tecnica. Mai utilizzare olio o altri lubrificanti, se non indicato espressamente nel manuale di istruzioni.
Come al solito, le raccomandazioni del manuale vanno lette e seguite con attenzione.
Ricorda: qualunque stampante funzionerà in modo corretto, purché sia regolata in maniera appropriata e a condizione che vengano effettuate con regolarità le adeguate operazioni di manutenzione richieste dall'unità.

I cavi di connessione

Ai cavi di collegamento vanno dedicate particolari cure ed attenzioni, essendo fondamentali per il buon funzionamento di tutto il sistema.
Poiché il punto più debole dei cavetti risulta essere quello dove il cavo termina nello spinotto, occorre essere

molto diligenti nell'osservare alcune semplici ed elementari precauzioni, deducibili comunque anche con un minimo di buon senso:
— evitare sempre e comunque di estrarre gli spinotti dalle prese tirandoli per il cavetto: questa operazione risulta spesso fatale per la continuità elettrica assicurata dallo spinotto;
— cercare, nei limiti del possibile, di far compiere ai cavetti poche pieghe brusche;
— limitare al massimo le operazioni di inserzione e disinserzione: infatti le prese e gli spinotti subiscono nel corso di queste manovre una sollecitazione che nel tempo tende a peggiorare il contatto elettrico. È senz'altro utile ripristinare di tanto in tanto il migliore contatto, agendo con delicatezza sullo spinotto mediante una piccola pinza;
— in caso di guasto la riparazione del cavo è molto semplice: basta individuare il punto di rottura (di solito cedono sempre le saldature) e con un piccolo saldatore rimettere a posto le cose. Evitare riparazioni diverse dalla saldatura (tipo giunzioni con nastro adesivo, viti, fili aggiunti, ecc.);
— controllare che il contatto avvenga sempre nel migliore dei modi, eliminando le eventuali tracce di ossido o sporcizia con uno straccetto imbevuto di sostanze solventi, tipo alcool o benzina.

LINGUAGGIO

Interpreti e compilatori

Quando scriviamo un programma in BASIC è facile dimenticare che in realtà il nostro computer non è in grado di capire quei comandi che noi gli impartiamo, ma può eseguire soltanto istruzioni date secondo una sequenza di numeri binari. Questi numeri, che costituiscono la lingua nativa del microprocessore del computer, vengono quindi forniti dall'interprete BASIC, che nel corso dell'esecuzione traduce una dopo l'altra le varie istruzioni componenti il programma.

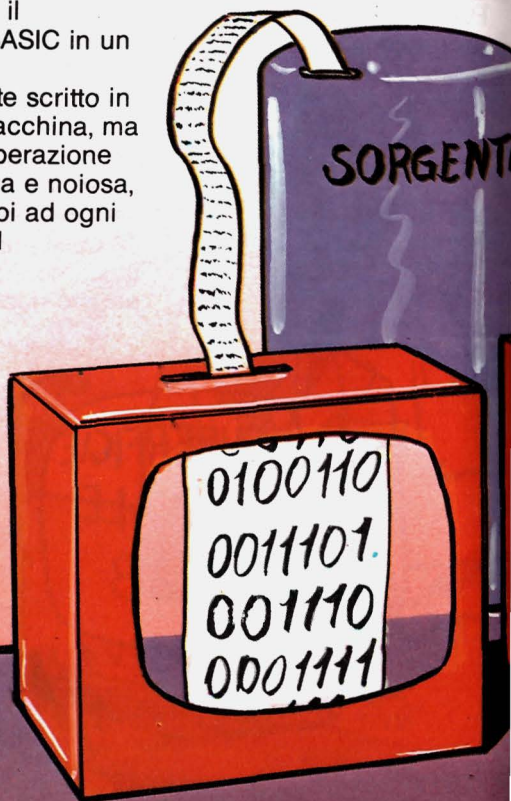
L'interprete BASIC nei confronti dell'elaboratore si comporta esattamente come un traduttore simultaneo, che decifra istantaneamente ciò che gli viene detto in frasi comprensibili ed eseguibili dalla CPU.

LINGUAGGIO

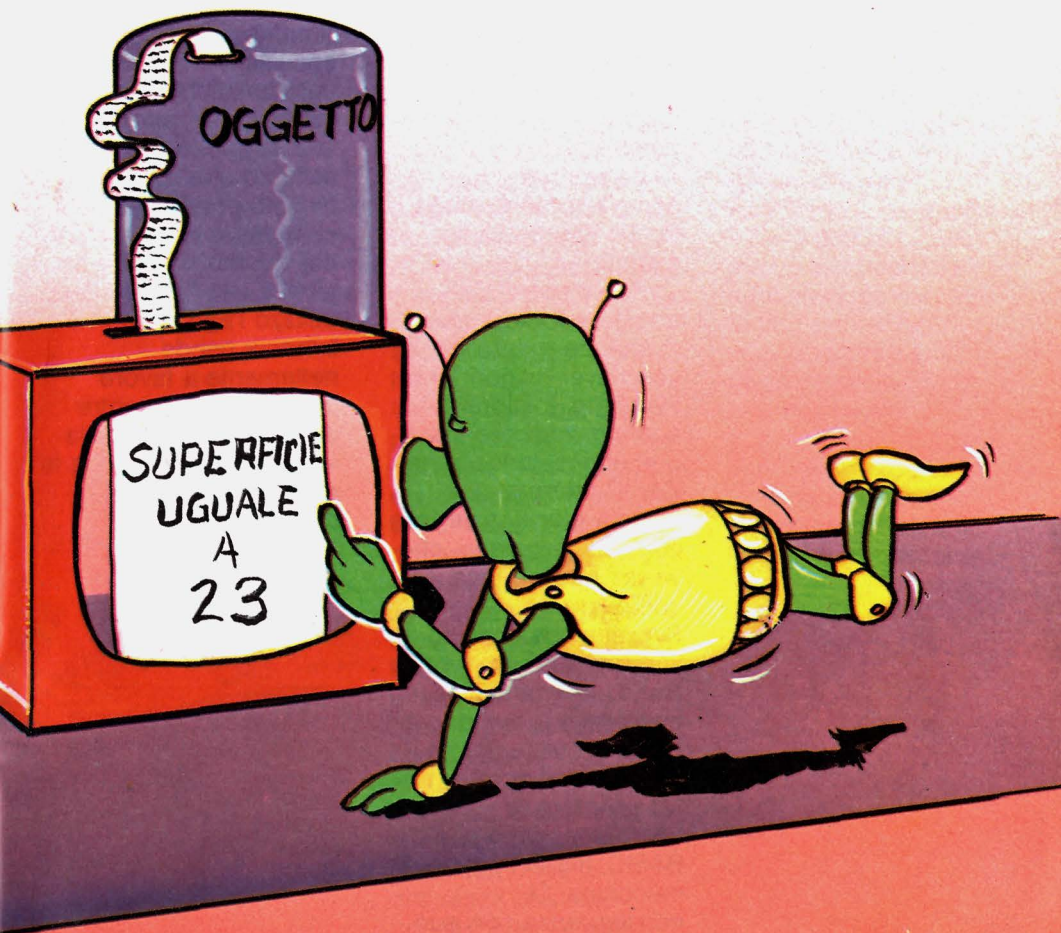
Agli effetti pratici possiamo pertanto immaginare l'interprete come una specie di scatola magica alla quale vengono fornite in entrata le istruzioni BASIC e che in uscita risponde con ordini "digeribili" dall'unità centrale.
In realtà l'interprete BASIC è anch'esso un programma vero e proprio, il cui compito è appunto quello di tradurre le istruzioni BASIC in istruzioni in linguaggio macchina.
Per quanto veloce ed ottimizzato possa essere, il programma traduttore richiede tuttavia un certo intervallo di tempo per poter svolgere il proprio lavoro: in altre parole, la traduzione da istruzioni BASIC a istruzioni in linguaggio macchina necessita di un certo lavoro e di conseguenza introduce un certo rallentamento nell'esecuzione.
La cosa può apparire alquanto ingarbugliata: perché ricorrere al traduttore quando si potrebbe scrivere il programma direttamente in istruzioni eseguibili dal microprocessore, ottenendo per giunta le risposte molto più in fretta? La risposta trova giustificazione nell'enorme sforzo che richiede - a confronto della programmazione in BASIC - la scrittura di programmi in linguaggio macchina.
Certamente i risultati e le risposte con la programmazione in assembler (così viene anche chiamato il linguaggio macchina) risultano molto più veloci e immediati, tuttavia il tempo necessario a risolvere e scrivere uno stesso programma in assembler e in BASIC è nettamente a favore del BASIC (mediamente il rapporto è di uno a venti).
L'interprete è quindi l'anello di congiunzione tra l'uomo e la macchina, cioè l'elemento che permette di realizzare una reciproca e perfetta comprensione.
Però, come detto prima, dato che l'interprete deve leggere ogni volta le varie istruzioni, analizzarle, verificare la correttezza della sintassi

ed eseguire le operazioni richieste, la velocità di lavoro non può che risentirne, risultando notevolmente inferiore a quella che il programma potrebbe avere se fosse scritto direttamente in linguaggio macchina.

Da queste considerazioni risulta chiaro che, per quanto di gran lunga superiore a quella di qualunque essere umano, la velocità di esecuzione di un programma BASIC non potrà mai superare un certo livello.
Si potrebbe pensare, per aumentarla, di tradurre manualmente il programma BASIC in un programma corrispondente scritto in linguaggio macchina, ma sarebbe un'operazione alquanto lunga e noiosa, da ripetersi poi ad ogni variazione del programma.

Le operazioni meccaniche, lunghe, noiose e ripetitive sono proprio il genere di cose che possiamo far eseguire a un calcolatore: basterà allora scrivere (una sola volta) un programma che traduca ciascuna istruzione BASIC in una

corrispondente routine in linguaggio macchina, e i nostri "problemi" saranno risolti. Un programma del genere si chiama compilatore BASIC: legge il nostro testo e, con pazienza, scrive (o, meglio, compila) un programma in linguaggio macchina che fa il medesimo lavoro.

Il primo vantaggio di un programma compilato è l'eliminazione del procedimento di interpretazione delle istruzioni. Inoltre, dato che la sintassi viene controllata durante la compilazione, si risparmia ulteriore tempo e, tra l'altro, non c'è più il rischio che il programma contenga errori di sintassi non rilevati.

Ma il vantaggio maggiore viene dal fatto che il compilatore, avendo la possibilità di leggere l'intero testo

BASIC, può eliminare le lente operazioni di ricerca delle linee (che l'interprete BASIC deve invece eseguire ad ogni GOTO o GOSUB) e delle variabili (per utilizzare una variabile occorre infatti che il traduttore compia un certo numero di operazioni, principalmente rivolte all'analisi della posizione occupata nella memoria da quella variabile).
Per tutti questi fattori la velocità di un programma compilato è notevolmente superiore a quella dello stesso programma interpretato.
Naturalmente, anche la compilazione ha i suoi svantaggi:
— il programma, di solito, aumenta di dimensioni;
— ci sono spesso istruzioni disponibili con l'interprete e non con il compilatore;
— il lavoro di compilazione richiede tempo (anche diversi minuti);
— non è possibile arrestare il programma compilato e riprenderlo dopo aver magari fatto stampare il valore di qualche variabile (cosa perfettamente lecita nel BASIC interpretato);
— ogni modifica va fatta sul testo originario, che va poi nuovamente compilato.
Si preferisce di solito scrivere il programma in BASIC normale (cioè interpretato), metterlo a punto e collaudarlo a fondo, e poi compilarlo.
Un programma compilato non è più in BASIC, ma in linguaggio macchina. Di conseguenza non può essere caricato in memoria con un semplice LOAD, o fatto partire con RUN, ma occorre usare altri comandi.
Da quello che abbiamo detto fino a questo punto sembrerebbe che un compilatore sia estremamente più vantaggioso di un interprete. In realtà, ognuno dei due presenta specifiche caratteristiche, alcune vantaggiose, altre svantaggiose, valutabili soltanto analizzando il singolo problema. Tieni comunque presente che per quanto riguarda la facilità e la flessibilità di utilizzo l'ago della bilancia pende nettamente a favore dell'interprete, mentre dal punto di vista della velocità di esecuzione le cose volgono in direzione del compilatore.

Rappresentazione dei numeri

Finora abbiamo sempre evitato qualsiasi cosa che non fosse aritmetica abbastanza semplice, e continueremo a fare così. Ma sarebbe sbagliato non dare un breve sguardo alle capacità aritmetiche del computer e soprattutto alla rappresentazione dei numeri nei computer.
L'elaborazione di un dato richiede infatti che lo stesso sia contenuto nella memoria dell'elaboratore in una forma immediatamente utilizzabile dall'unità centrale. Poiché l'unica rappresentazione riconoscibile dalla CPU è quella digitale, anche i numeri devono quindi essere espressi in tale forma.
Conosci già come si fa a passare dalla notazione decimale a quella binaria e viceversa, e quindi non ci soffermeremo a parlare di questo argomento.
Esamineremo invece alcuni aspetti piuttosto interessanti che emergono nell'uso pratico dei numeri col computer.
Innanzitutto, il principale consiglio è quello di esaminare sempre con una certa dose di diffidenza qualsiasi numero che sia stato utilizzato all'interno di un programma. La ragione di questo suggerimento è che i numeri all'interno del computer, a causa della conversione da decimali a binari, subiscono quasi sempre arrotondamenti o troncamenti, che ne modificano - per quanto leggermente - l'esatto valore. Per esempio, nella nostra familiare notazione decimale ti ricorderai certamente che un numero come 1/3 o 1/7 non è in grado di avere un'espressione esatta: 1/3=0.333333..... Noi diamo per scontato che aggiungendo al numero tanti 3 quanti ne vogliamo possiamo ottenere un grado di approssimazione accettabile in qualsiasi particolare problema. Nel sistema binario, allo stesso modo, alcuni numeri non possono

essere espressi esattamente; per esempio, nella forma decimale possiamo dire che 1/10=0.1, ma quando il numero viene cambiato in forma binaria non può essere rappresentato con la dovuta precisione.
Per verificarlo con mano è sufficiente che tu batta e faccia eseguire questo breve programma:

```
10 LET I=1
20 LET I=I-0.1
30 IF I=0 THEN END
40 PRINT I
50 GOTO 20
```

Teoricamente tutto è esatto. Queste istruzioni costituiscono un ciclo che dovrebbe sottrarre per 10 volte la quantità 0.1 al numero 1. La riga 30 indica infatti al computer di arrestarsi quando la variabile I vale 0.
Come invece avrai facilmente modo di constatare, la condizione I=0 non verrà mai raggiunta, e il tuo VIC 20 proseguirà all'infinito a sottrarre ad I il numero 0.1. I risultati delle varie operazioni compariranno via via sullo schermo, facendoti vedere immediatamente la causa di questo comportamento, cioè l'approssimazione introdotta nel risultato dai vari calcoli.
In generale è sempre meglio diffidare delle cifre meno significative (cioè le cifre all'estrema destra) di qualsiasi risultato numerico: potrebbero essere conseguenza di imprecisioni interne.
Per funzionare come volevamo il nostro programma avrebbe dovuto essere modificato in questa riga

```
30 IF I<0.1 THEN END
```

e tutto sarebbe andato a posto.
Abbiamo quindi accertato che i numeri all'interno della memoria possono subire delle alterazioni.
Esistono infatti dei limiti al numero di cifre che un computer è in grado di trattare: numeri troppo grandi o troppo piccoli devono essere quasi necessariamente arrotondati. Vediamo meglio questo fatto.
Un numero molto grande (oppure molto piccolo) può essere rappresentato mediante la cosiddetta notazione scientifica, cioè con una forma abbreviata del numero stesso, nella quale si indica la potenza di 10 alla quale il numero va elevato, preceduta dalla lettera "E" (in pratica, di quante cifre va spostata la virgola). La parte a

sinistra delle E si chiama mantissa, la successiva esponente:

Forma normale	Notazione scientifica
1000000	1E+06
0.00000001	1E-07
0.000586321	0.586321E-3
93000000000	0.93E+11

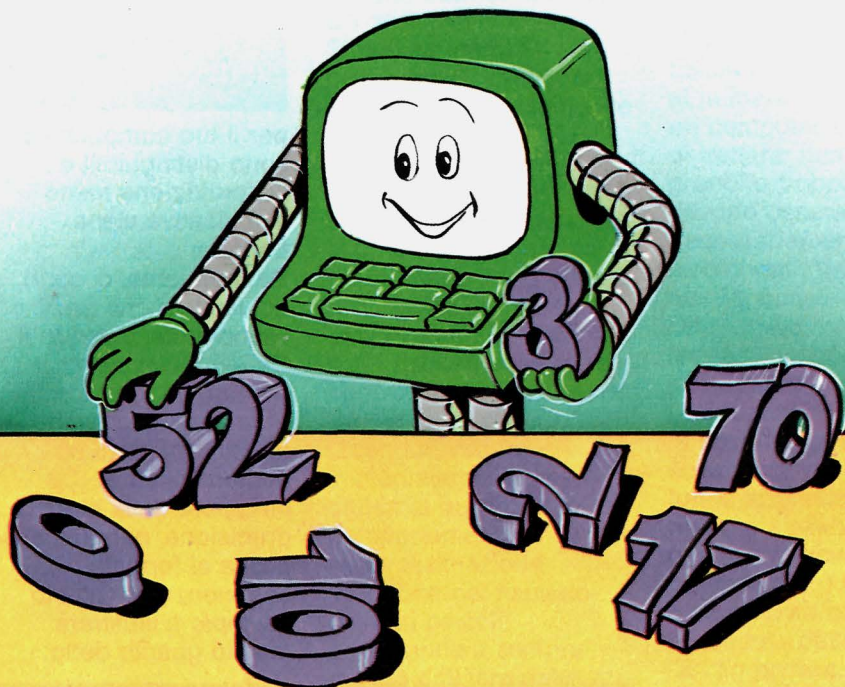
Il calcolatore, al momento dell'introduzione dei numeri in forma normale, lì trasformerà internamente in questo formato (chiamato anche in virgola mobile), consistente nel porre il punto decimale dopo la prima cifra significativa e aggiustare di conseguenza l'esponente.
Naturalmente, se la mantissa supera una certa lunghezza alcune cifre vanno perdute.
Così, i due numeri

12.00000001
12.00000008

per il tuo computer non sono distinguibili e l'informazione meno significativa viene perduta.
Naturalmente, ci sono dei limiti anche per l'esponente. Quando il BASIC stampa un numero, lo fa in formato normale, a meno che questo comporti un numero di cifre maggiore della sua precisione, nel qual caso passa al formato esponenziale. Questo esempio ti illustrerà meglio quanto detto:

```
10 PRINT "NUMERO", "RAPPRESENTAZIONE"
20 PRINT
30 FOR I=−10 TO 10
40 PRINT "10↑"; I, 10↑I
50 NEXT I
```

Funzioni numeriche

Ormai abbiamo quasi terminato l'esame delle varie funzioni disponibili sul tuo VIC 20: infatti, ce ne restano ancora soltanto due. Tra poco vedremo tuttavia come sia possibile, attraverso un utile e potente strumento fornito dal BASIC, creare e definire a proprio piacere qualsiasi altra funzione, che non faccia già parte del gruppo inserito nella macchina sin dalla fabbrica.

EXP

La funzione esponenziale viene utilizzata per elevare il numero "e" a una certa potenza. Essa calcola cioè il valore della costante e (base dei logaritmi cosiddetti naturali) elevato alla potenza fornita dall'argomento specificato.
Il numero e è un particolare valore, che viene definito matematicamente in una forma molto precisa, ma di cui il tuo VIC 20 utilizza una approssimazione pari a 2.7182818.
Vediamola subito al lavoro:

```
PRINT EXP(1)
```

dice al computer di visualizzare il risultato di e elevato alla 1. Sullo schermo apparirà quindi il valore 2.7182818.
L'origine di questo valore è abbastanza complessa; questo breve programma ti illustrerà tuttavia come esso possa essere ottenuto in forma approssimata:

```
10 FOR I=1 TO 1000
20 LET E=(1+1/I)↑I
30 PRINT E, EXP(1)
40 NEXT I
```

Sulla sinistra dello schermo apparirà infatti una serie di numeri, che all'aumentare di I tenderà ad essere sempre più prossima al valore di e fornito dalla funzione EXP (visibile sulla destra).

Sintassi della funzione

EXP (espressione)

LOG

La funzione LOG si utilizza invece per compiere l'operazione esattamente opposta a quella eseguita con EXP. LOG calcola infatti il logaritmo naturale, cioè in base e, dell'argomento.
Così, se e è il risultato di EXP(1), LOG(e) risulterà 1. Gli argomenti si scambiano con i risultati e viceversa.
Il logaritmo è una funzione particolarmente utile e preziosa in numerosi calcoli matematici, poiché permette di abbreviare conti ed operazioni, grazie ad alcune importanti proprietà di cui essa gode.
Soprattutto nella tecnica e nei calcoli scientifici è quindi di basilare importanza disporre dell'uso dei logaritmi.
Se comunque non ne conosci le regole di utilizzo, non ti preoccupare: visto che non ne hai avuto bisogno fino ad ora, forse i logaritmi non ti serviranno neppure in futuro. Nel caso tu comunque desiderassi usarli lo stesso (magari per generare strani numeri o cose del

genere), ricordati di non assegnare mai a LOG un argomento negativo o nullo. Proprio come la funzione SQR, anche LOG non può lavorare su numeri minori di zero (e nemmeno sullo zero).

Sintassi della funzione

LOG (espressione)

DEF FN

Capita spesso che nello stesso programma siano presenti calcoli od espressioni simili; in tal caso è possibile scriverli una sola volta, per poi richiamarli con una subroutine.
Queste ultime hanno però lo svantaggio di lavorare con i nomi di variabili specificate nella routine stessa.
Ad esempio, se volessimo approssimare all'intero più vicino il valore di un certo numero, potremmo scrivere una cosa del genere:

```
100 REM APPROSSIMA ALL'INTERO
110 REM PIU' VICINO IL VALORE DI X
120 X=INT(X+.5)
130 RETURN
```

Questa subroutine fornisce però soltanto la media dei valori contenuti nella sola variabile X.
Potrebbe darsi che nel nostro programma avessimo bisogno di arrotondare anche i valori di P e di Q. Per usare la subroutine dovremmo riassegnare i valori delle variabili ogni volta:

```
20 LET X=P : GOSUB 100 : LET P=X
30 LET X=Q : GOSUB 100 : LET Q=X
```

In tutte le funzioni presenti sul VIC 20 si fa però riferimento alle variabili interessate, scrivendo quindi SQR(A) per ottenere la radice quadrata di A e SQR(Z) per ricavare quella di Z.

Definendo una nuova funzione, potremmo analogamente specificare la variabile su cui fare le operazioni. Il BASIC consente al programmatore di definire nuove funzioni matematiche, utilizzabili in seguito come se facessero parte del linguaggio stesso. Per definire una nuova funzione bisogna inserire nel programma una istruzione del tipo

```
DEF FN nome (argomento)=espressione
```

Nel caso del nostro esempio la soluzione avrebbe quindi potuto essere:

```
10 DEF FN R(X)=INT(X+.5)
```

Da quel punto in avanti la funzione sarebbe stata considerata come definita e il computer l'avrebbe eseguita in tutta tranquillità.
L'istruzione DEF FN consente quindi di definire nuove funzioni, oltre a quelle facenti parte del linguaggio vero e proprio.
Approfondiamo meglio la sintassi da utilizzare:
— DEF FN è una parola riservata, che indica all'elaboratore di definire una nuova funzione;
— nome è la parola che verrà impiegata nel resto del programma per chiamare la funzione definita con DEF FN; essa deve rispettare le stesse regole di formazione dei nomi delle variabili numeriche decimali;
— argomento è la variabile che serve per definire gli operandi sui quali interverrà la funzione;
— espressione indica invece le operazioni che il computer dovrà eseguire per giungere al risultato.
Dopo che il programma ha eseguito la linea

```
DEF FN R(X)=INT(X+.5)
```

la funzione R(X) diventa quindi una parola riservata del BASIC e può essere usata come fosse parte del linguaggio.
Per esempio:

```
300 A=FN R(C)
```

vuol dire: calcola il valore di A, applicando al valore di C la funzione R definita prima.

La X, che abbiamo usato alla linea 10 per definire la funzione, viene sostituita con il valore della espressione che segue FN R, cioè con il valore di C. Si dice quindi che X è un parametro formale (o parametro fittizio), cioè ha il solo scopo di indicare dove va impiegato l'argomento della funzione quando questa viene chiamata. Naturalmente, perché il BASIC possa riconoscerla, la definizione della funzione (cioè l'esecuzione della linea di programma che la definisce) deve avvenire prima che la funzione possa essere richiamata. Qui sotto puoi trovare una serie di funzioni matematiche che non sono comprese nel BASIC standard, ma che è possibile definire mediante DEF FN nel momento in cui esse si rendessero eventualmente necessarie:

Funzione	Descrizione
SEC(X)=1/COS(X)	Secante
CSC(X)=1/SIN(X)	Cosecante
COT(X)=1/TAN(X)	Cotangente
ARCSIN(X)=ATN(X/SQR(-X*X+1))	Arcoseno (Seno inverso)
ARCCOS(X)=-ANT(X/SQR(-X*X+1))+1.5708	Arcocoseno (Coseno inverso)
ARCSEN(X)=ATN(SQR(X*X-1))+(SGN(X)-1)*1.5708	Arcotangente (Tangente inversa)
ARCCSC(X)=ATN(1/SQR(X*X-1))+(SGN(X)-1)*1.5708	Cosecante inversa
ARCCOT(X)=-ATN(X)+1.5708	Cotangente inversa
SINH(X)=(EXP(X)-EXP(-X))/2	Seno iperbolico
COSH(X)=(EXP(X)+EXP(-X))/2	Coseno iperbolico
TANH(X)=-EXP(-X)/(EXP(X)+EXP(-X))*2+1	Secante iperbolica
SECH(X)=2/(EXP(X)+EXP(-X))	Tangente iperbolica
CSCH(X)=2/(EXP(X)-EXP(-X))	Cosecante iperbolica
COTH(X)+EXP(-X)/(EXP(X)-EXP(-X))*2+1	Cotangente iperbolica
ARGSINH(X)=LOG(X+SQR(X*X+1))	Seno iperbolico inverso
ARGCOSH(X)=LOG(X+SQR(X*X-1))	Coseno iperbolico inverso
ARGTANH(X)=LOG((1+X)/(1-X))/2	Tangente iperbolica inversa
ARGSECH(X)=LOG((SQR(-X*X+1)+1)	Secante iperbolica inversa
ARGCSCH(X)=LOG(SGN(X)*SQR(X*X+1)+1)/X	Cosecante iperbolica inversa
ARGCOTH(X)=LOG((X+1)/(X-1))/2	Cotangente iperbolica inversa
MOD(A)=INT((A/B-INT(A/B))*B+.05)*SGN(A/B)	Modulo
LOGB(X)=LOG(X)/LOG(B)	Logaritmo in base B

Esempi

```
10 DEF FN A(R)=R*R*TT
20 LET X=2
30 PRINT FN A(X)
```

Definisce una funzione (*il cui nome* è A) *avente* come scopo quello di calcolare l'area del cerchio di raggio R. R è il parametro formale. La linea 20 calcola l'area del cerchio di raggio 2. X costituisce invece il parametro reale.

```
10 DEF FN SEN(X)=SIN(X)
```

Definisce una nuova funzione, chiamata S, che esegue la stessa operazione della funzione intrinseca SIN.

```
40 PRINT FN SEN(3)
50 PRINT SIN(3)
```

L'effetto di queste istruzioni sarà quindi la stampa di due valori assolutamente identici.

```
90 DEF FN A(X)=PEEK(2040)+PEEK(2041)*256
```

Questa funzione fornisce *come risultato un valore* indipendente dall'argomento utilizzato. È infatti perfettamente lecito (sempre rispettando le regole imposte dall'istruzione) definire delle funzioni che, nonostante lo richiedano, non utilizzano l'argomento all'interno dell'espressione. In tal caso è addirittura possibile non inserire alcun argomento, lasciando così le parentesi vuote. Avremmo così potuto

anche scrivere DEF FN A()=PEEK(2040)+PEEK (2041)*256 e il risultato non sarebbe cambiato.

```
20 DEF FN SOM(X)=3+SIN(2.3)
30 PRINT FN SOM(41)
40 PRINT FN SOM(3.82)
```

Anche in questo caso l'argomento non avrà alcuna influenza sul risultato, visto che non compare nell'espressione. Le due istruzioni di stampa, nonostante richiamino funzioni con argomento differente, produrranno quindi risultati identici.

Sintassi della funzione

DEF FN nome funzione (variabile [, variabile]) = espressione

Usare e definire le funzioni

La nostra parte di lezione dedicata alla programmazione ti illustrerà alcuni tra i moltissimi possibili utilizzi ai quali si presta utilmente la definizione delle funzioni.
Il primo esempio calcola il valore finale di un deposito bancario mantenuto in un libretto di risparmio per un certo numero di anni. Per esempio, supponendo di depositare 5000000 in un conto corrente per quattro anni e di disporre di un interesse annuo del 10% calcolato trimestralmente (cioè gli interessi vengono capitalizzati 4 volte all'anno), è possibile sapere quale sarà la cifra a nostra disposizione al termine dei 4 anni.
Vanno specificati in ingresso: la somma depositata, il numero di periodi di calcolo dell'interesse, il tasso di interesse annuo e il tempo di deposito espresso in anni. In uscita risulterà il capitale al termine dell'investimento.
Nel programma abbiamo definito (linea 110) una funzione per arrotondare al valore intero più vicino la somma risultante alla fine del deposito.
Ecco il listato

```
 10 REM
 20 N$="VALORE FINALE DI UN DEPOSITO"
 30 REM
 40 PRINT CHR$(147)
 50 PRINT N$:PRINT
 60 INPUT "DEPOSITO INIZIALE";DEP
 70 INPUT "PERIODI ANNUI";PER
 80 INPUT "INTERESSE (%)";IN
 90 INPUT "NUMERO DI ANNI";ANNI
100 PRINT:PRINT "VALORE FINALE";
110 DEF FN ARR(X)=INT(X*100+.5)/100
120 A=DEP
130 FOR I=1 TO PER*ANNI
140 A=A+A*IN/100/PER
150 NEXT I
160 A=FN ARR(A)
170 PRINT A
180 END
```

Il secondo programma calcola invece il tempo necessario perché un capitale investito raddoppi di valore. Così, supponendo che la nostra banca calcoli gli interessi quattro volte all'anno e paghi il 12%

di interesse annuo, quanti anni ci vorranno per raddoppiare un investimento di 3000000?
Il programma fornirà immediatamente la risposta, una volta che gli saranno stati specificati in input il tasso di interesse annuo e il numero di periodi di calcolo dell'interesse (introducendo 12 significherà per esempio che l'interesse dovrà essere calcolato mensilmente).
In questo caso la definizione di funzione viene usata per semplificare l'espressione del calcolo del tempo di raddoppio, la cui formula è facilmente reperibile in qualsiasi libro di ragioneria.
Ecco il listato:

```
 10 REM
 20 N$ ="TEMPO DI RADDOPPIO"
 30 REM
 40 PRINT CHR$(147)
 50 PRINT N$:PRINT
 60 INPUT "PERIODI ANNUI";PER
 70 INPUT "INTERESSE ANNUO";IN
 80 PRINT:PRINT "TEMPO DI RADDOPPIO (ANNI)";
 90 DEF FN RAD(X)=LOG(2)/(X*LOG(1+IN/100/X))
100 DEF FN ARR(X)=INT(X*100+.5)/100
110 Y= FN RAD(PER)
120 Y= FN ARR(Y)
130 PRINT Y
140 END
```

L'ultimo programma utilizza la DEF FN per simulare un tavolo da gioco, precisamente di roulette.
Alla riga 60 puoi infatti vedere questa istruzione:

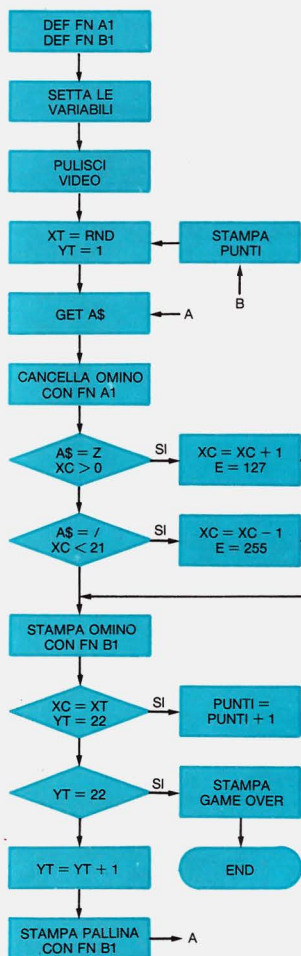
```
60 DEF FN PALL(X) = INT (RND(0) * 37)
```

che consente al tuo VIC 20 di estrarre numeri casuali compresi tra 0 e 36, cioè quelli tipici del "tavolo verde".
La cosa interessante dell'istruzione è però un'altra: come puoi facilmente osservare, nell'espressione che definisce la funzione (cioè nella parte posta a destra del segno di uguale) non compare l'argomento della funzione. In altre parole - quando la funzione sarà utilizzata - il valore assunto dall'argomento non avrà alcuna importanza ai fini del calcolo della funzione stessa, ed i valori casuali saranno estratti senza avere alcun riferimento con esso.
Tanto vale quindi utilizzare come argomento sempre lo

stesso valore (per esempio zero): proprio quello che è stato fatto alla linea 110. Questa è comunque una regola valida in generale: è quindi perfettamente lecito definire funzioni che non utilizzano come dato in ingresso l'argomento loro fornito (anche se in molti casi la cosa può non avere molto senso).
Ecco il listato:

```
10 REM
20 N$="ROULETTE"
30 REM
40 PRINT CHR$(147)
50 PRINT N$:PRINT
60 DEF FN PALL(X)=INT(RND(0)*37)
70 INPUT "QUAL'È IL TUO CAPITALE";CAP:GOTO 80
75 PRINT "SEI A QUOTA";CAP
80 INPUT "SU QUALE NUMERO VUOI PUNTARE";NUM
90 INPUT "QUANTO PUNTI";PUN
100 IF PUN>CAP THEN PRINT "TROPPO!! RIPETI.":GOTO 90
110 US=FN PALL(0)
115 PRINT "È USCITO IL";US
120 IF NUM=US THEN CAP=CAP+36*PUN:PRINT "COMPLIMENTI!!":
    GOTO 140
130 CAP=CAP-PUN
140 IF CAP<=0 THEN PRINT "MI DISPIACE, HAI SBALLATO.":GOTO 180
150 INPUT "ANCORA (S/N)";R$
160 IF R$="N" THEN 180
170 GOTO 75
180 END
```

Videogame

Utilizziamo alcune funzioni definibili per realizzare un semplice gioco. Devi aiutare un omino in fondo allo schermo a raccogliere le palline rosse che cadono dall'alto. Ogni pallina presa ti permetterà di ottenere un punto.

```
10 DEF FN A 1 (A) = A + XC : DEF FN B 1 (A)
   = A + XT + YT * 22
15 LET A = 8164 : LET B = 38884 : LET C =
   7680 : LET D = 38400
20 LET XC = 10 : LET E = 127 : LET P = 0 :
   POKE 650, 128 : PRINT "[♥]"
25 LET XT = INT (RND (1) * 21) : LET YT = 1
30 GET A$ : IF A$ = "" THEN GOTO 50
35 POKE FN A 1 (A), 32
40 IF A$ = "/" AND XC < 21 THEN LET XC = XC
   + 1 : LET E = 255
45 IF A$ = "Z" AND XC > 0 THEN LET XC = XC
   - 1 : LET E = 127
50 POKE FN A 1 (A), E: POKE FN A 1 (B), 0
55 IF YT = 22 AND XT = XC THEN GOSUB 80 :
   GOTO 25
60 IF YT = 22 THEN GOTO 85
65 POKE FN B 1 (C), 32 : LET YT = YT + 1
70 POKE FN B 1 (C), 81 : POKE FN B 1 (D), 2
75 GOTO 30
80 LET P = P + 1 : PRINT "[S][←] PUNTI"; P :
   RETURN
85 PRINT SPC (94) "[▓] GAME OVER" : POKE 650, 0
90 INPUT "[←][Q][]][]][]][]] GIOCHI ANCORA"; R$
95 IF R$ = "S" THEN RUN
```

Nella linea 10 la funzione A1 viene utilizzata per muovere il personaggio e B1 per muovere la pallina.
Nella linea 20 si assegnano le posizioni iniziali sullo schermo sia del personaggio che della pallina.
La linea 25 genera un numero casuale tra 0 e 21 che individua la colonna percorsa dalla pallina.
Le linee dalla 30 alla 50 controllano e gestiscono le posizioni del personaggio.
La 65 e la 70 controllano la discesa della pallina.
La 55 e la 60 verificano se la pallina è sfuggita (nel qual caso il gioco termina) oppure se è stata presa.
La linea 80 conta le palline prese.
Dalla linea 85 alla 95 termina il gioco e viene offerta la possibilità di ricominciare.

VIDEOESERCIZI

La funzione definita nel programma seguente esegue i calcoli necessari per visualizzare una parabola. Definiscine una per ottenere la rappresentazione di un grafico diverso.

```
 10 PRINT CHR$ (147)
 20 GOSUB 300
 30 DEF FN X (Y) = Y * Y/4
 35 PRINT "PARABOLA"
 40 FOR I = - 9 TO 9
 50 PRINT TAB (FN X (I)); "*"
 60 NEXT
 70 GOSUB 500
 80 END
300 FOR I = 1 TO 10
310 PRINT CHR$ (221)
320 NEXT
330 FOR I = 1 TO 21
340 PRINT CHR$ (192);
350 NEXT
355 PRINT
360 FOR I = 1 TO 10
370 PRINT CHR$ (221)
380 NEXT
390 PRINT CHR$ (19)
400 RETURN
500 GET A$
510 IF A$ = "" THEN GO TO 500
520 RETURN
```